Saturation NOW

Ming-Ying Chung and Gianfranco Ciardo

Department of Computer Science and Engineering

University of California at Riverside

Riverside, CA 92521, USA

{chung,ciardo}@cs.ucr.edu

Work supported in part by NASA grant NAG-1-02095 and NSF grants CCR-0219745 and ACI-0203971

Outline

- **Introduction**
  - *Parallel and distributed state-space generation*
- **Background**
  - *MDDs* encoding of the state-space
  - *Kronecker* encoding of the next state function
  - *Saturation-style* fixed point iteration strategy
- **Saturation NOW** (Network of workstation)
  - MDDs distribution, canonicity, and cache management
  - **Saturation NOW in action**
- **Dynamic memory load balancing**
  - Pairwise memory load balancing dilemmas
  - *Nested* memory load balancing approach
- **Experimental results and conclusions**

Introduction

Parallel and distributed state-space generation

- *Formal verification* : for *quality assurance*
- *Model checking* : a, model base, automatic verification approach

  E. Clarke and E. Emerson. *Synthesis of synchronization skeletons for branching time temporal logic*, Logic of Programs 1981
- *State-space generation* : the first step in model checking (memory-intensive)
- *Binary decision diagrams* (BDDs) : *symbolic* state-space construction

  R. Bryant, *Graph-based algorithm for boolean function manipulation*, IEEE TC 1986
- *Parallel and distributed model checking* :

  uses computation resources on a network or different computer architecture
- *Saturation NOW* : a *message-passing* scheme based on *saturation*

  G. Ciardo, G. Lüettgen, and R. Siminiceanu, *Saturation : an efficient iteration strategy for symbolic state-space generation*, TACAS 2001

Related parallel and distributed work

- *Explicit* work : high memory consumption

  D. Nicol and G. Ciardo, *Automated parallelization of discrete state-space generation*, Journal of Parallel and Distributed Computing 1997

  U. Stern and D. Dill, *Parallelizing the Murphy verifier*, CAV 1997

- *Symbolic* work on *shared-memory multi-processor* : special HW

  S. Kimura and E. M. Clarke, *A parallel algorithm for constructing BDDs*, ICCD 1990

- *Symbolic* work on *distributed shared memory* : special SW

  Y. Parasuram, E. Stabler, and S.-K. Chin, *Parallel implementation of BDD algorithm using a DSM*, HICSS 1994

- *Symbolic* work on NOW

  - Job-based slicing : overlap image computation

    K. Milvang-Jensen and A. Hu, *BDDNOW*, FMCAD 1998

    O. Grumberg, T. Heyman, and A. Schuster, *A work-efficient distributed algorithm for reachability analysis*, CAV 2003

  - Level-based slicing : scaling problem, no memory load balancing

    R. Ranjan, J. Snaghavi, R. Brayton, and A. Sangiovanni-Vincentelli, *BDDs on NOWs*, ICCD 1996

Background

Structured discrete-state models

- A *structured discrete state model* is a triple $(\widehat{\mathcal{S}}, \mathcal{S}^{init}, \mathcal{N})$ where
  - $\widehat{\mathcal{S}}$ is the set of *potential states* of the model
  - $\mathcal{S}^{init} \subseteq \widehat{\mathcal{S}}$ is the set of *initial states*
  - $\mathcal{N} : \widehat{\mathcal{S}} \rightarrow 2^{\widehat{\mathcal{S}}}$ is the *next-state* function

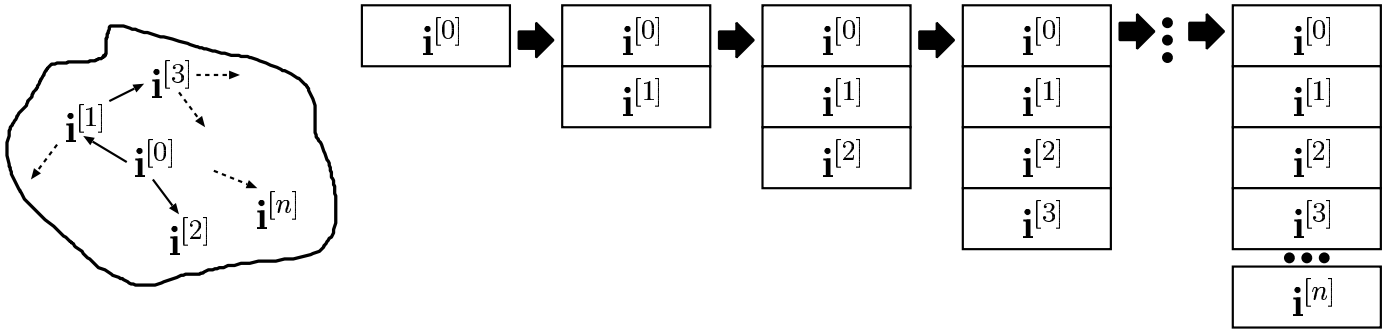
- The *reachable state-space* $\mathcal{S} \in \widehat{\mathcal{S}}$ is the smallest set
  - containing $\mathcal{S}^{init}$
  - closed with respect to $\mathcal{N}$

$$\mathcal{S} = \mathcal{S}^{init} \cup \mathcal{N}(\mathcal{S}^{init}) \cup \mathcal{N}^2(\mathcal{S}^{init}) \cup \mathcal{N}^3(\mathcal{S}^{init}) \cup \cdots = \mathcal{N}^*(\mathcal{S}^{init})$$

Explicit vs. symbolic state space generation

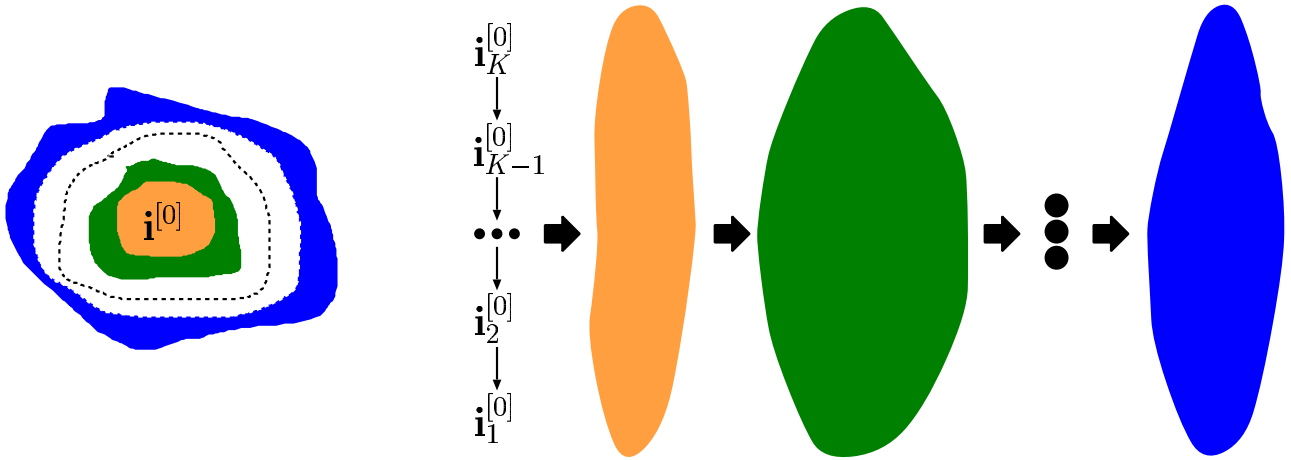
Explicit approach adds *one state* at a time

- memory O(states), increases linearly, peaks at the end



Symbolic approach with decision diagrams adds *sets of states* at a time

- memory O(decision diagram nodes), grows and shrinks



(Quasi-reduced ordered) MDDs

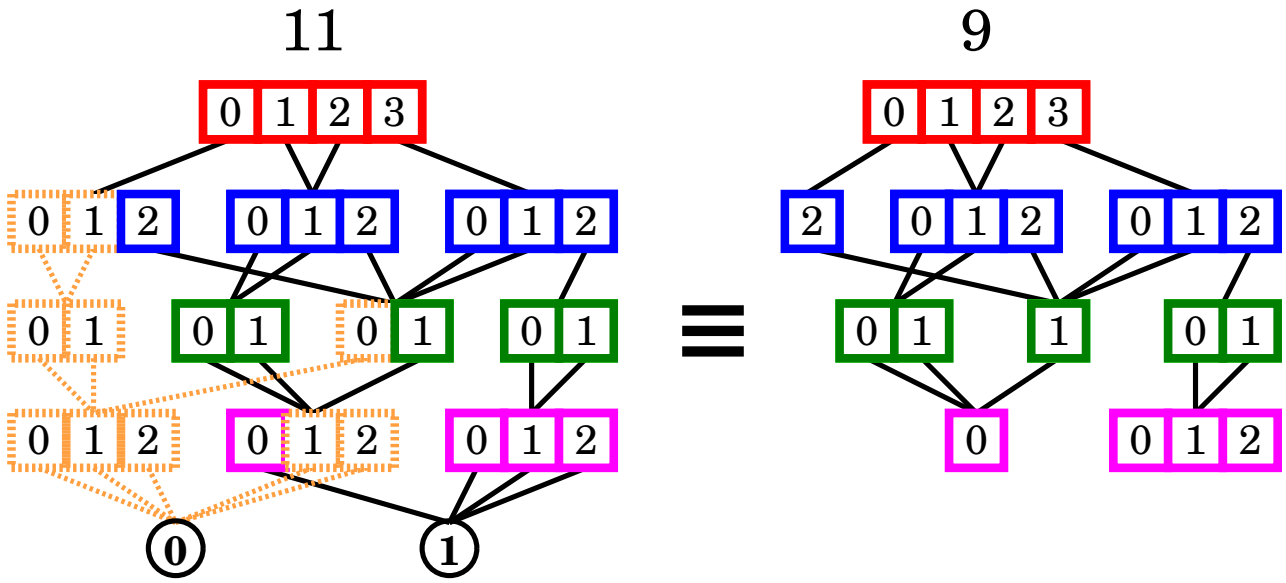
- We structure the states *by level*
- We do not store those paths that *point to the terminal node* 0

$\mathcal{S}_4 = \{0, 1, 2, 3\}$

$\mathcal{S}_3 = \{0, 1, 2\}$

$\mathcal{S}_2 = \{0, 1\}$

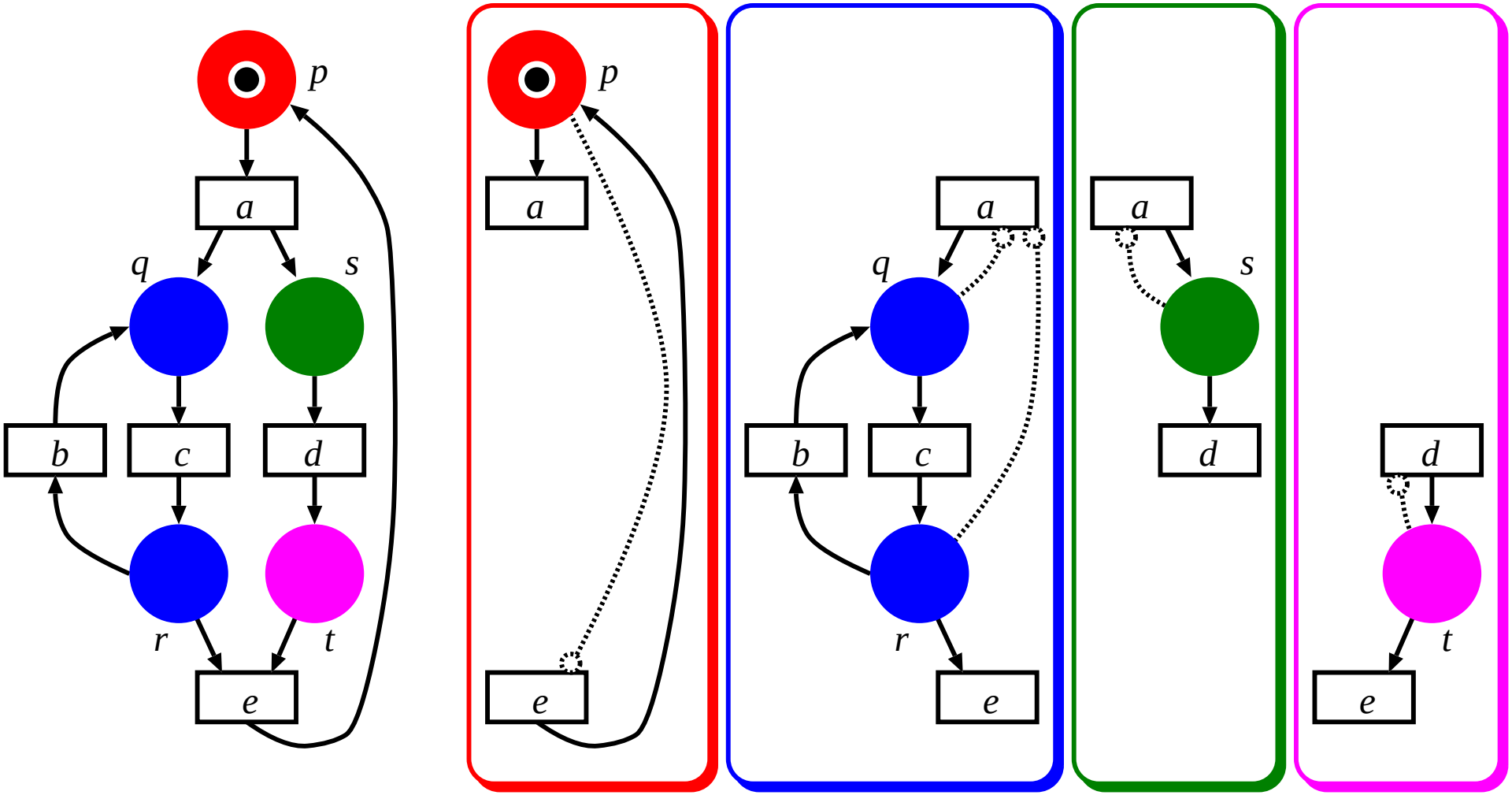
$\mathcal{S}_1 = \{0, 1, 2\}$



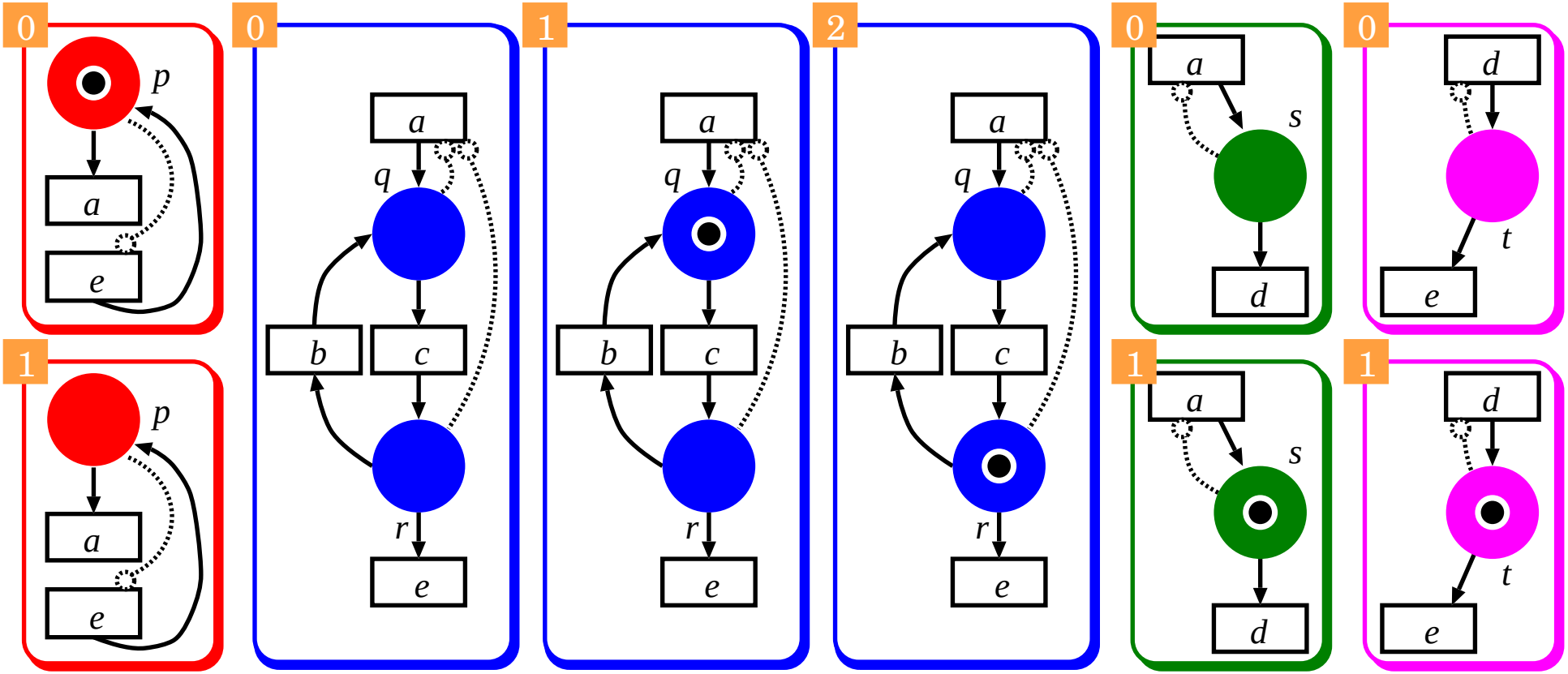
$$\mathcal{S} = \underbrace{\left\{ \begin{matrix} 0 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 \\ 2 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 2 & 2 & 2 & 2 & 2 & 2 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 1 & 2 \end{matrix} \right\}}_{19}$$

Model decomposition

- A bounded model could be decomposed into unbounded submodels
- We modify each submodel with *inhibit arcs* to enforce *known bounds*



Local states of each submodel



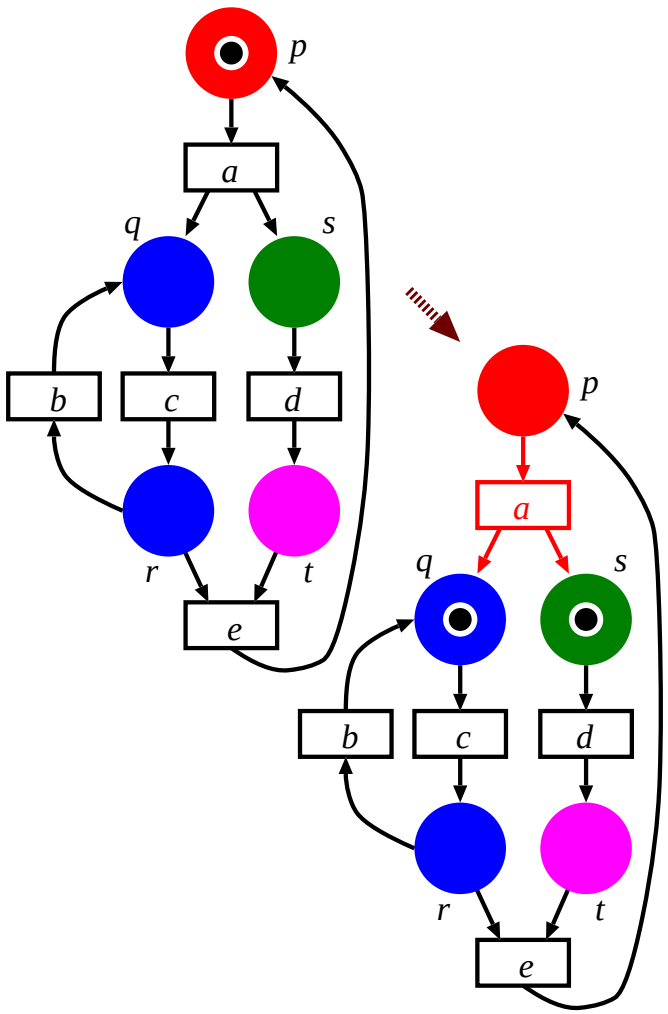
$$\mathcal{S}_4 = \{p^1, p^0\} \equiv \{0,1\}$$

$$\mathcal{S}_3 = \{q^0 r^0, q^1 r^0, q^0 r^1\} \equiv \{0,1,2\}$$

$$\mathcal{S}_2 = \{s^0, s^1\} \equiv \{0,1\}$$

$$\mathcal{S}_1 = \{t^0, t^1\} \equiv \{0,1\}$$

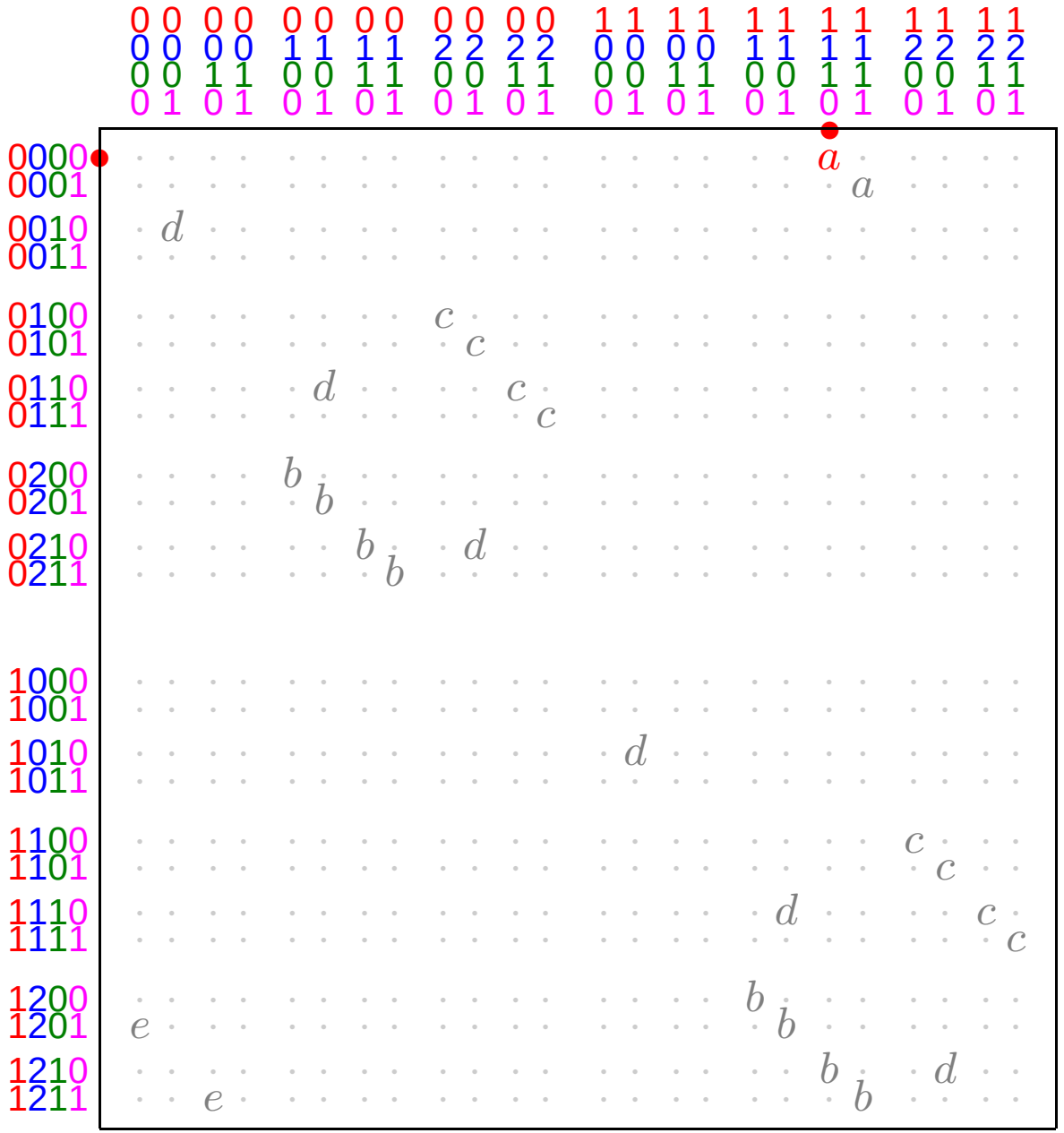
The state transition matrix $\mathbf{N}$ $(K = 4)$



$$(p^1, q^0 r^0, s^0, t^0) = (0, 0, 0, 0)$$

$$\Downarrow$$

$$(p^0, q^1 r^0, s^1, t^0) = (1, 1, 1, 0)$$



Saturation

An MDD node p at level k is ***saturated*** if the state set it encodes is a fixed point w.r.t. any event α s.t. $Top(\alpha) \leq k$ (thus, all nodes below p are also saturated)

- build the MDD encoding of initial state set

  (if there is a single initial state, there is one node per level)

- From $k=1$ to K, saturate each node at level k by

  firing all events α s.t. $Top(\alpha) = k$

  (if this creates nodes below level k, saturate them immediately upon creation)

- When the root node is saturated, all reachable states have been discovered

States are not discovered in breadth-first order

Enormous time and memory efficiency compared to traditional iteration

Saturation NOW

Saturation NOW

- **Problem** :

  We target relatively large models where state-space generation with a single workstation must rely on virtual memory or run out of memory

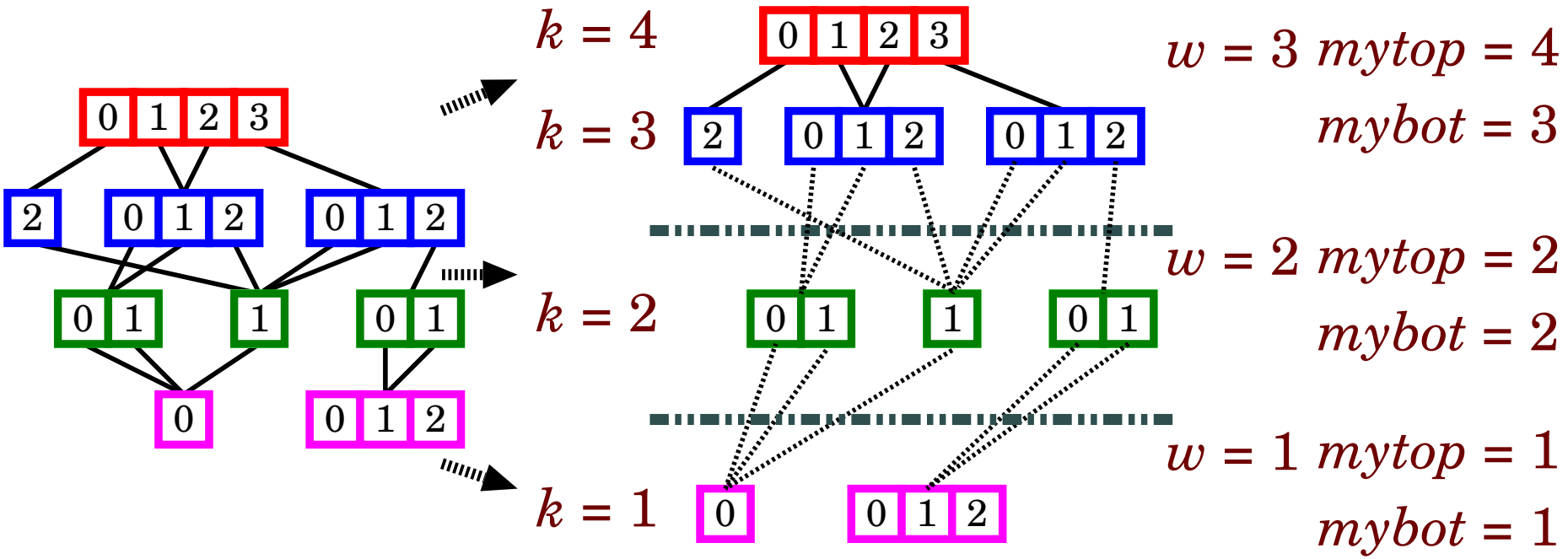
- **Solution** :

  We distribute the saturation algorithm on a NOW to increase the available memory at the cost of an acceptable communication overhead

  - Only *one* workstation is *active* at any time
  - The goal of this algorithm is not a theoretical speedup
  - Balancing memory load requires negligible memory overhead and achieves *nearly ideal memory scalability*

Level-based distributed MDD allocation

- We number $W \leq K$ workstations in the NOW from W down to 1
- For $W \geq w \geq 1$, w owns MDD levels from $mytop$ to $mybot$
  - $mytop = \lfloor w \cdot K/W \rfloor$
  - $mybot = \lfloor (w-1) \cdot K/W \rfloor + 1$



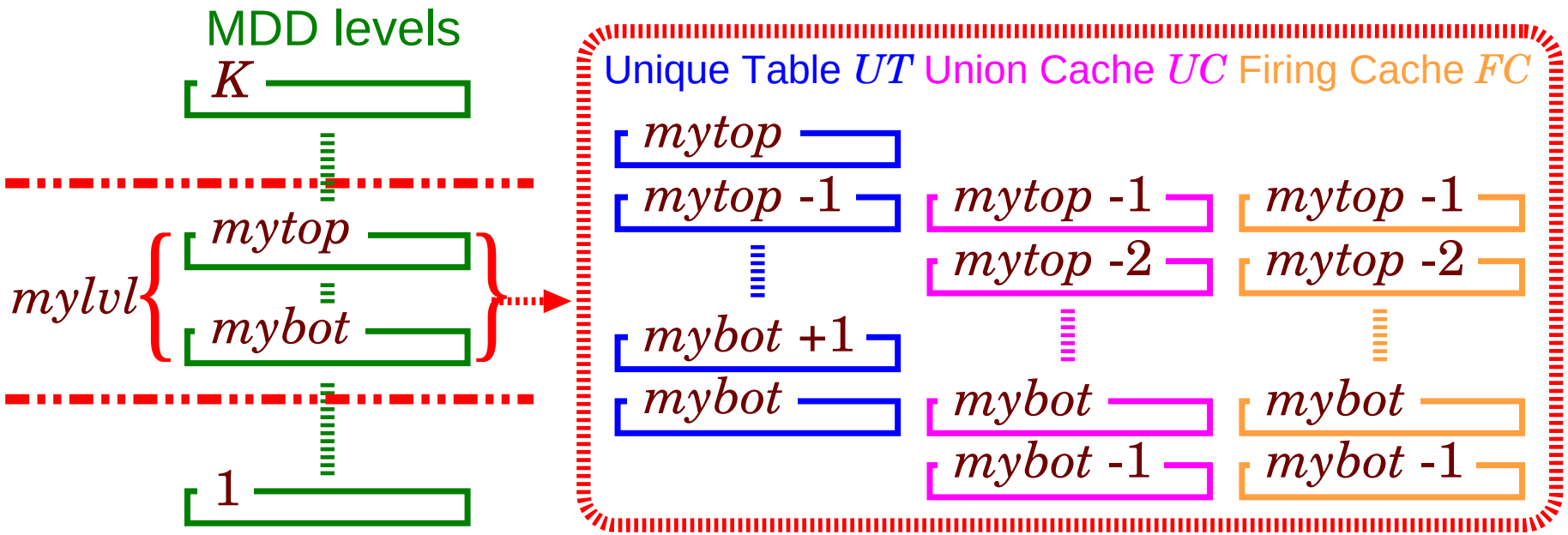
Distributed unique table and caches allocation

- *Unique tables*

  level-based hash tables to retain canonicity

- *Firing caches* and *Union caches*

  level-based integer caches to avoid duplicate operations



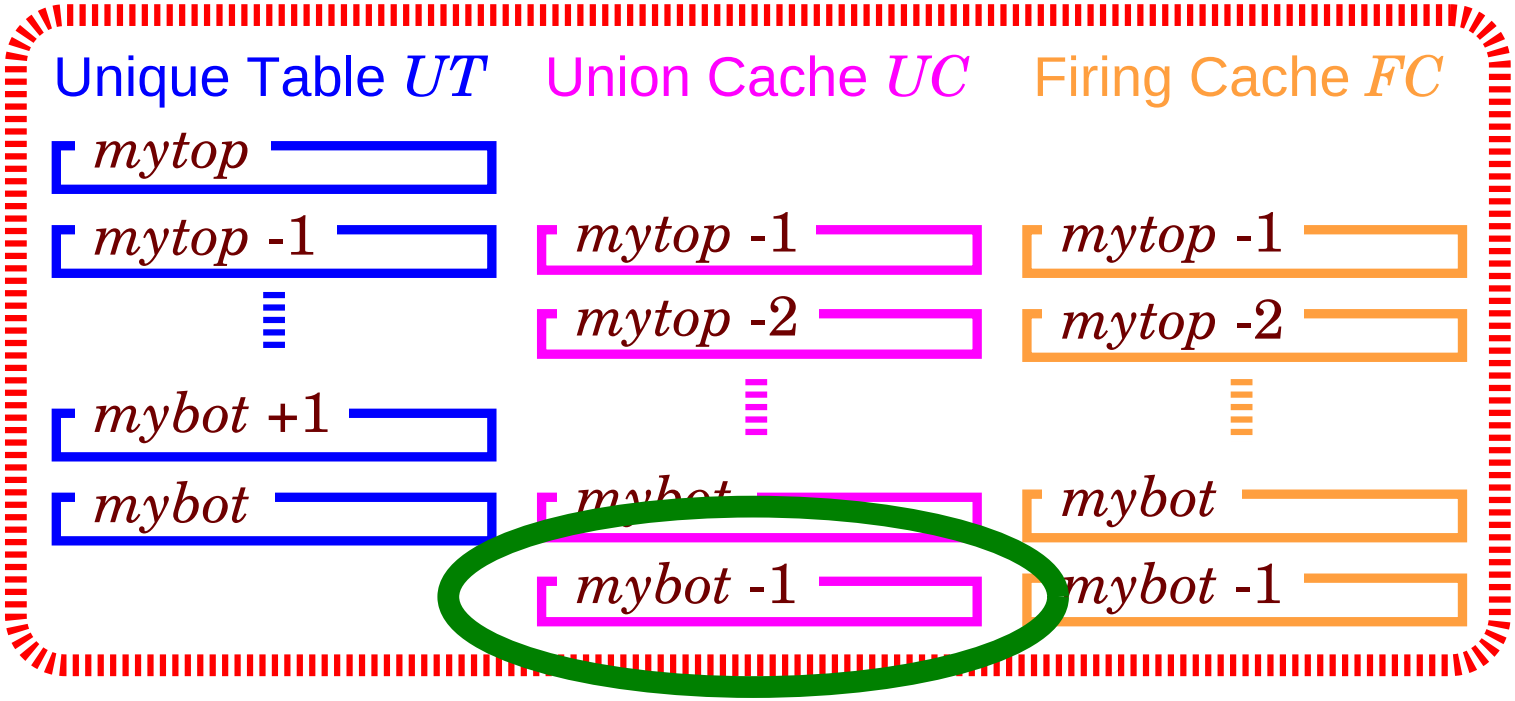
Union Heuristic

- *Union cache prediction* heuristic

$$Union(p,q) = r \;\Rightarrow\; Union(p,r) = Union(q,r) = r$$

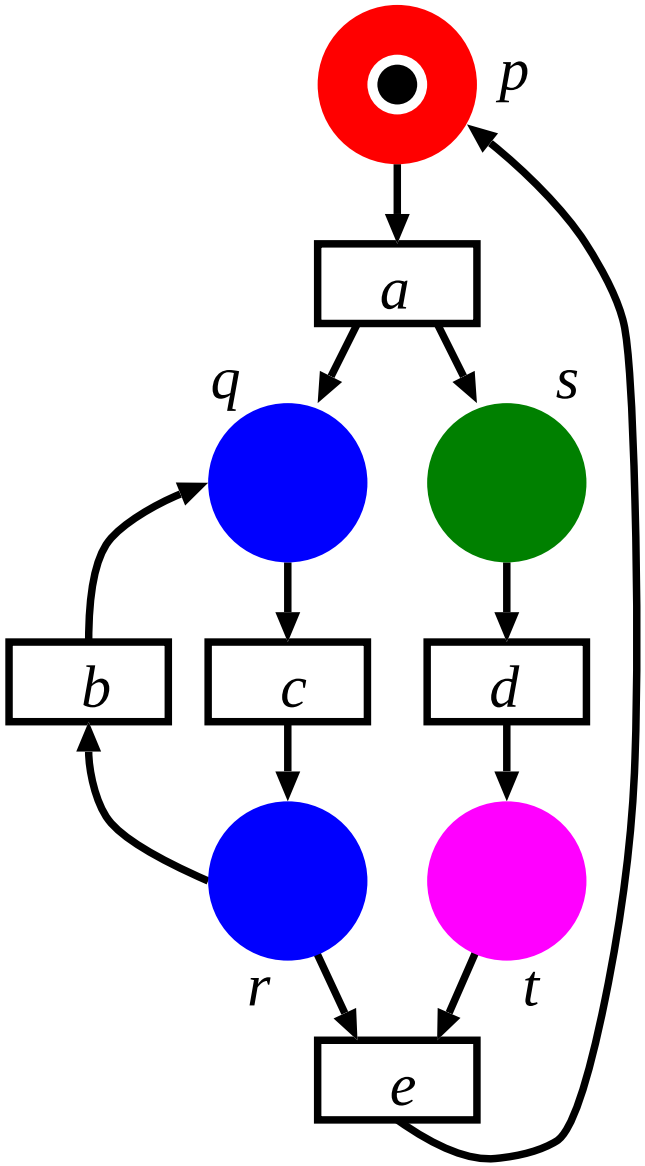
  - Reduces the runtime of SMART by up to 12% in some benchmarks
  - We employ this heuristics only for level $mybot - 1$ of each $w > 1$

    (this is where it saves both computation and communication)



Saturation NOW in Action

The Kronecker matrix $\mathbf{N}$ $(K = 4)$



$\mathcal{S}_4 = \{p^1, p^0\} \equiv \{0, 1\}$ $\Rightarrow w_3$

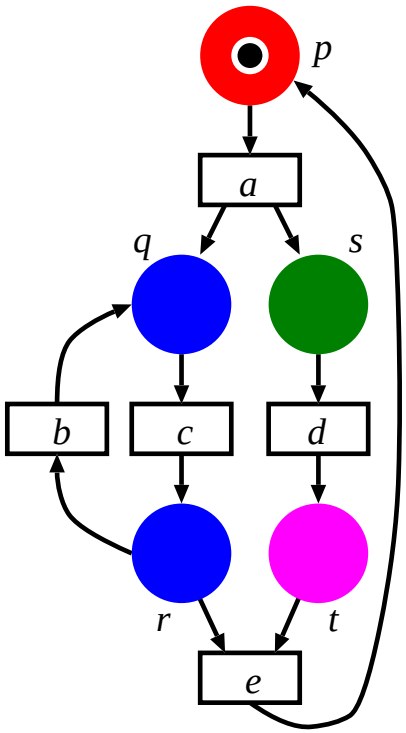
$\mathcal{S}_3 = \{q^0 r^0, q^1 r^0, q^0 r^1\} \equiv \{0, 1, 2\}$ $\Rightarrow w_3$

$\mathcal{S}_2 = \{s^0, s^1\} \equiv \{0, 1\}$ $\Rightarrow w_2$

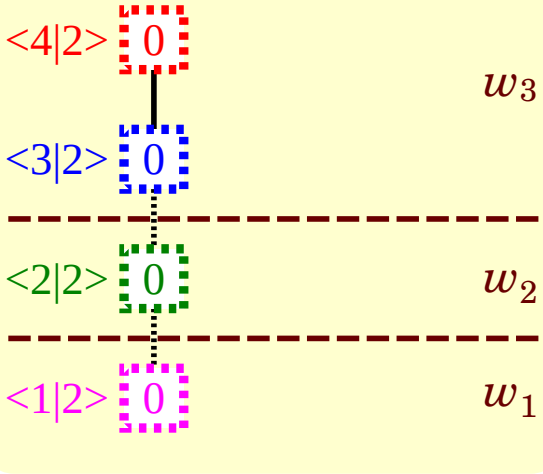
$\mathcal{S}_1 = \{t^0, t^1\} \equiv \{0, 1\}$ $\Rightarrow w_1$

a	$\{b, c\}$	d	e
$0 : 1$ $1 : -$	**I**	**I**	$0 : -$ $1 : 0$
$0 : 1$ $1 : -$ $2 : -$	$0 : -$ $1 : 2$ $2 : 1$	**I**	$0 : -$ $1 : -$ $2 : 0$
$0 : 1$ $1 : -$	**I**	$0 : -$ $1 : 0$	**I**
I	**I**	$0 : 1$ $1 : -$	$0 : -$ $1 : 0$

Create initial state on each $1 \leq w \leq W$ where $|\mathcal{S}^{init}| = 1$



a	$\{b, c\}$	d	e
$0 : 1$ $1 : -$	**I**	**I**	$0 : -$ $1 : 0$
$0 : 1$ $1 : -$ $2 : -$	$0 : -$ $1 : 2$ $2 : 1$	**I**	$0 : -$ $1 : -$ $2 : 0$
$0 : 1$ $1 : -$	**I**	$0 : -$ $1 : 0$	**I**
I	**I**	$0 : 1$ $1 : -$	$0 : -$ $1 : 0$



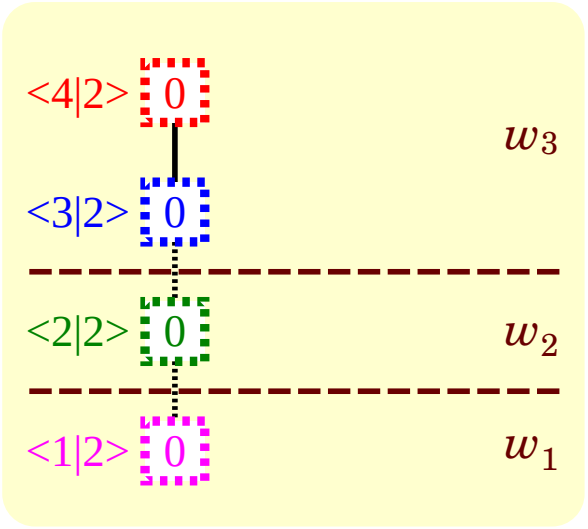
$\mathcal{S}_4 = \{\boxed{p^1}, p^0\} \equiv \{\boxed{0}, 1\}$

$\mathcal{S}_3 = \{\boxed{q^0 r^0}, q^1 r^0, q^0 r^1\} \equiv \{\boxed{0}, 1, 2\}$

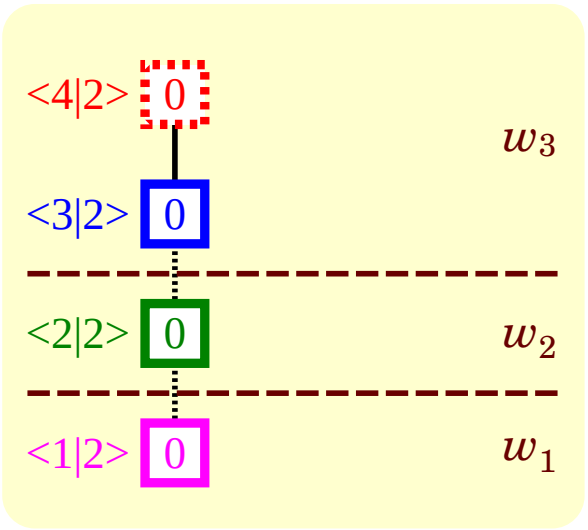
$\mathcal{S}_2 = \{\boxed{s^0}, s^1\} \equiv \{\boxed{0}, 1\}$

$\mathcal{S}_1 = \{\boxed{t^0}, t^1\} \equiv \{\boxed{0}, 1\}$

Saturate $\langle 1|2\rangle$ on w_1, $\langle 2|2\rangle$ on w_2, $\langle 3|2\rangle$ on w_3 : no firing



a	$\{b, c\}$	d	e
$0 : 1$ $1 : -$	**I**	**I**	$0 : -$ $1 : 0$
$0 : 1$ $1 : -$ $2 : -$	$0 : -$ $1 : 2$ $2 : 1$	**I**	$0 : -$ $1 : -$ $2 : 0$
$0 : 1$ $1 : -$	**I**	$0 : -$ $1 : 0$	**I**
I	**I**	$0 : 1$ $1 : -$	$0 : -$ $1 : 0$



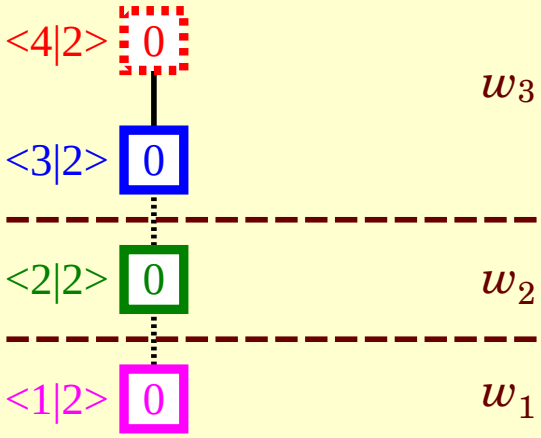
$\mathcal{S}_4 = \{\boxed{p^1}, p^0\} \equiv \{\boxed{0}, 1\}$

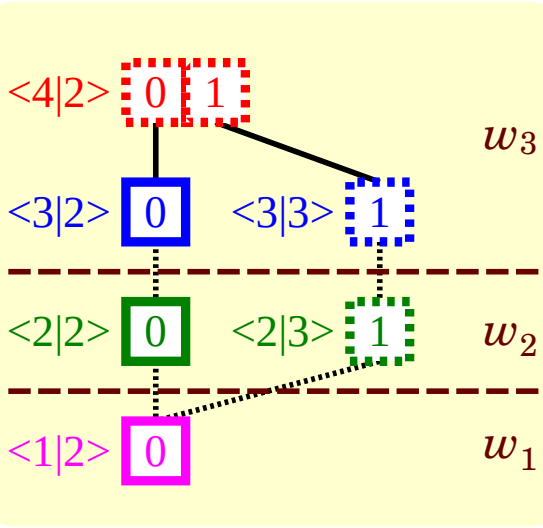
$\mathcal{S}_3 = \{\boxed{q^0 r^0}, q^1 r^0, q^0 r^1\} \equiv \{\boxed{0}, 1, 2\}$

$\mathcal{S}_2 = \{\boxed{s^0}, s^1\} \equiv \{\boxed{0}, 1\}$

$\mathcal{S}_1 = \{\boxed{t^0}, t^1\} \equiv \{\boxed{0}, 1\}$

Saturate $\langle 4|2 \rangle$ on w_3 : local and remote firing of a





a	$\{b, c\}$	d	e
$0 : 1$ $1 : -$	**I**	**I**	$0 : -$ $1 : 0$
$0 : 1$ $1 : -$ $2 : -$	$0 : -$ $1 : 2$ $2 : 1$	**I**	$0 : -$ $1 : -$ $2 : 0$
$0 : 1$ $1 : -$	**I**	$0 : -$ $1 : 0$	**I**
I	**I**	$0 : 1$ $1 : -$	$0 : -$ $1 : 0$

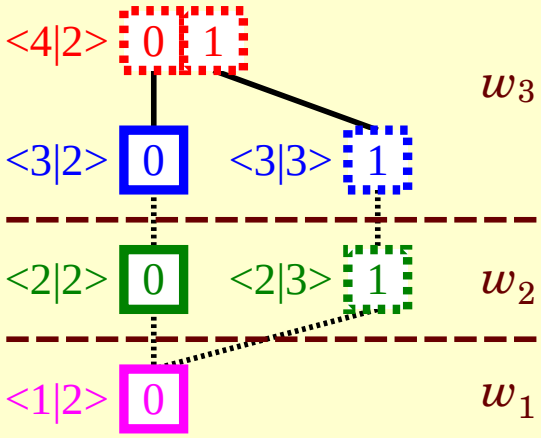
$$\mathcal{S}_4 = \{\boxed{p^1}, \boxed{p^0}\} \equiv \{\boxed{0}, \boxed{1}\}$$

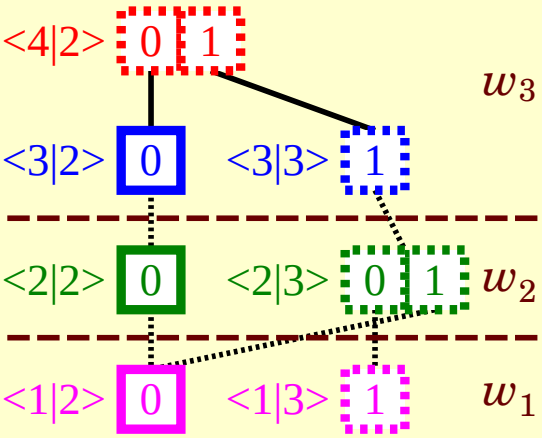
$$\mathcal{S}_3 = \{\boxed{q^0 r^0}, \boxed{q^1 r^0}, q^0 r^1\} \equiv \{\boxed{0}, \boxed{1}, 2\}$$

$$\mathcal{S}_2 = \{\boxed{s^0}, \boxed{s^1}\} \equiv \{\boxed{0}, \boxed{1}\}$$

$$\mathcal{S}_1 = \{\boxed{t^0}, t^1\} \equiv \{\boxed{0}, 1\}$$

Saturate $\langle 2|3 \rangle$ on w_2 : local and remote firing of d





a	$\{b, c\}$	d	e
$0 : 1$ $1 : -$	**I**	**I**	$0 : -$ $1 : 0$
$0 : 1$ $1 : -$ $2 : -$	$0 : -$ $1 : 2$ $2 : 1$	**I**	$0 : -$ $1 : -$ $2 : 0$
$0 : 1$ $1 : -$	**I**	$0 : -$ $1 : 0$	**I**
I	**I**	$0 : 1$ $1 : -$	$0 : -$ $1 : 0$

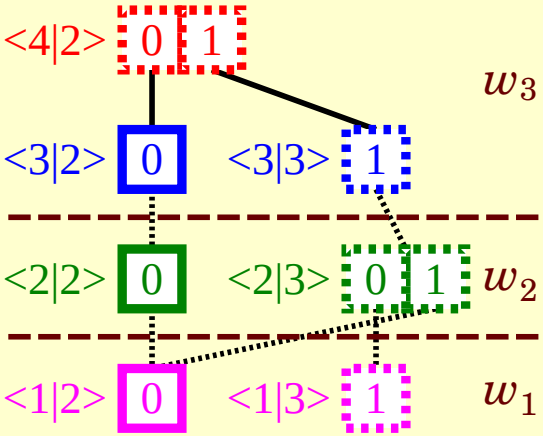
$$\mathcal{S}_4 = \{\boxed{p^1}, \boxed{p^0}\} \equiv \{\boxed{0}, \boxed{1}\}$$

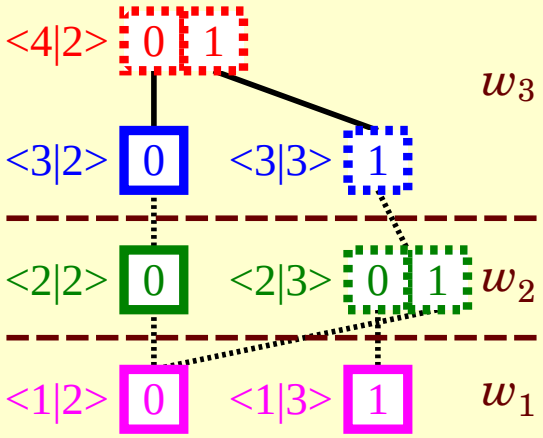
$$\mathcal{S}_3 = \{\boxed{q^0 r^0}, \boxed{q^1 r^0}, q^0 r^1\} \equiv \{\boxed{0}, \boxed{1}, 2\}$$

$$\mathcal{S}_2 = \{\boxed{s^0}, \boxed{s^1}\} \equiv \{\boxed{0}, \boxed{1}\}$$

$$\mathcal{S}_1 = \{\boxed{t^0}, \boxed{t^1}\} \equiv \{\boxed{0}, \boxed{1}\}$$

Saturate $\langle 1|3 \rangle$ on w_1 : no firing





a	$\{b, c\}$	d	e
$0 : 1$ $1 : -$	**I**	**I**	$0 : -$ $1 : 0$
$0 : 1$ $1 : -$ $2 : -$	$0 : -$ $1 : 2$ $2 : 1$	**I**	$0 : -$ $1 : -$ $2 : 0$
$0 : 1$ $1 : -$	**I**	$0 : -$ $1 : 0$	**I**
I	**I**	$0 : 1$ $1 : -$	$0 : -$ $1 : 0$

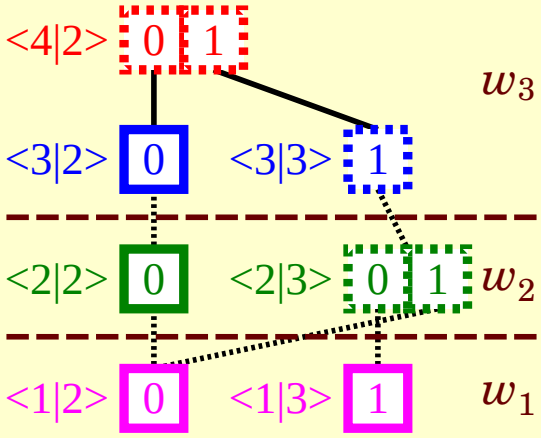
$\mathcal{S}_4 = \{\boxed{p^1}, \boxed{p^0}\} \equiv \{\boxed{0}, \boxed{1}\}$

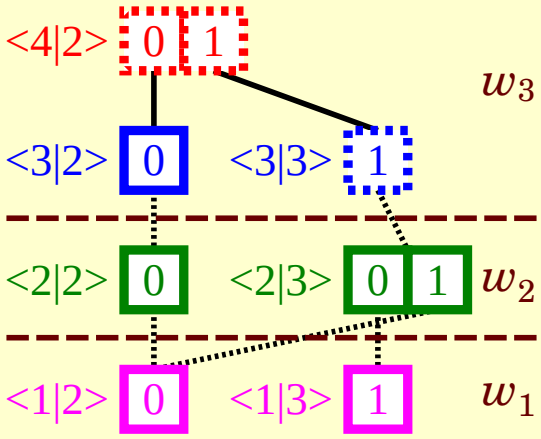
$\mathcal{S}_3 = \{\boxed{q^0 r^0}, \boxed{q^1 r^0}, q^0 r^1\} \equiv \{\boxed{0}, \boxed{1}, 2\}$

$\mathcal{S}_2 = \{\boxed{s^0}, \boxed{s^1}\} \equiv \{\boxed{0}, \boxed{1}\}$

$\mathcal{S}_1 = \{\boxed{t^0}, \boxed{t^1}\} \equiv \{\boxed{0}, \boxed{1}\}$

Each event α s.t. $Top(\alpha)=2$ is fired : $\langle 2|3\rangle$ is saturated





a	$\{b,c\}$	d	e
$0:1$ $1:-$	**I**	**I**	$0:-$ $1:0$
$0:1$ $1:-$ $2:-$	$0:-$ $1:2$ $2:1$	**I**	$0:-$ $1:-$ $2:0$
$0:1$ $1:-$	**I**	$0:-$ $1:0$	**I**
I	**I**	$0:1$ $1:-$	$0:-$ $1:0$

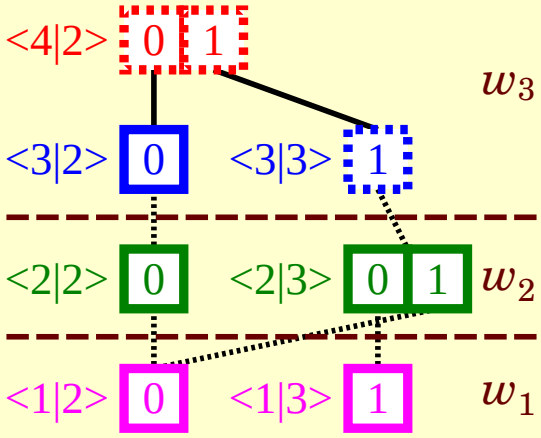
$\mathcal{S}_4 = \{\boxed{p^1}, \boxed{p^0}\} \equiv \{\boxed{0}, \boxed{1}\}$

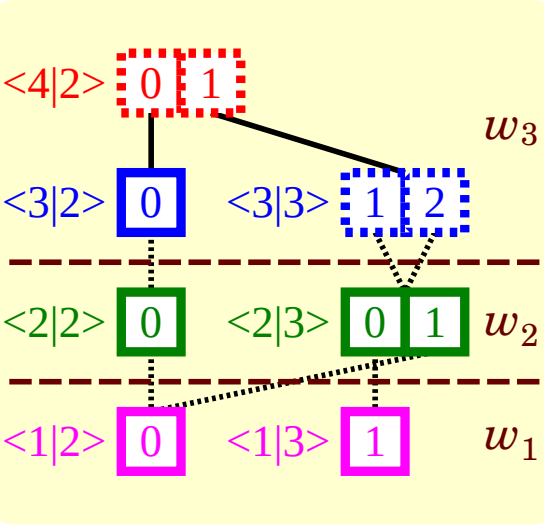
$\mathcal{S}_3 = \{\boxed{q^0r^0}, \boxed{q^1r^0}, q^0r^1\} \equiv \{\boxed{0}, \boxed{1}, 2\}$

$\mathcal{S}_2 = \{\boxed{s^0}, \boxed{s^1}\} \equiv \{\boxed{0}, \boxed{1}\}$

$\mathcal{S}_1 = \{\boxed{t^0}, \boxed{t^1}\} \equiv \{\boxed{0}, \boxed{1}\}$

Saturate $\langle 3|3\rangle$ on w_3 : local firing of $\{b, c\}$





a	$\{b, c\}$	d	e
$0 : 1$ $1 : -$	**I**	**I**	$0 : -$ $1 : 0$
$0 : 1$ $1 : -$ $2 : -$	$0 : -$ $1 : 2$ $2 : 1$	**I**	$0 : -$ $1 : -$ $2 : 0$
$0 : 1$ $1 : -$	**I**	$0 : -$ $1 : 0$	**I**
I	**I**	$0 : 1$ $1 : -$	$0 : -$ $1 : 0$

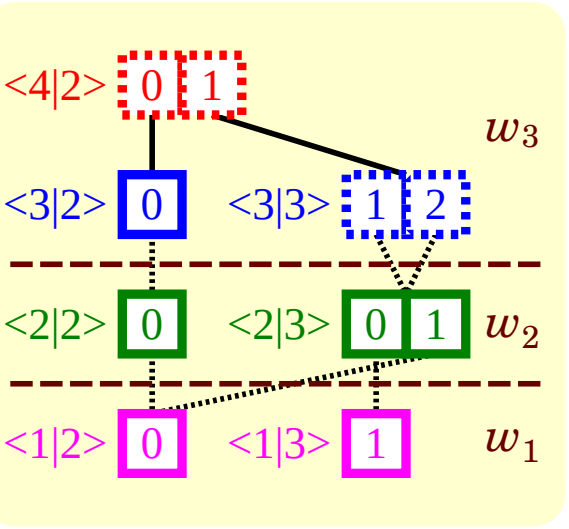
$$\mathcal{S}_4 = \{\boxed{p^1}, \boxed{p^0}\} \equiv \{\boxed{0}, \boxed{1}\}$$

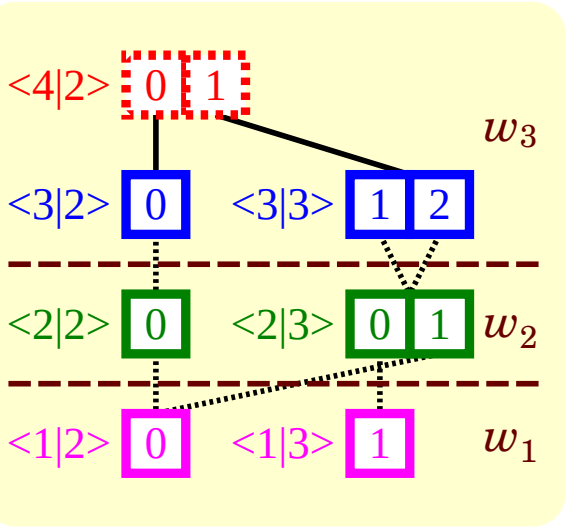
$$\mathcal{S}_3 = \{\boxed{q^0 r^0}, \boxed{q^1 r^0}, \boxed{q^0 r^1}\} \equiv \{\boxed{0}, \boxed{1}, \boxed{2}\}$$

$$\mathcal{S}_2 = \{\boxed{s^0}, \boxed{s^1}\} \equiv \{\boxed{0}, \boxed{1}\}$$

$$\mathcal{S}_1 = \{\boxed{t^0}, \boxed{t^1}\} \equiv \{\boxed{0}, \boxed{1}\}$$

Each event α s.t. $Top(\alpha)=3$ is fired : $\langle 3|3\rangle$ is saturated





a	$\{b,c\}$	d	e
$0:1$ $1:-$	**I**	**I**	$0:-$ $1:0$
$0:1$ $1:-$ $2:-$	$0:-$ $1:2$ $2:1$	**I**	$0:-$ $1:-$ $2:0$
$0:1$ $1:-$	**I**	$0:-$ $1:0$	**I**
I	**I**	$0:1$ $1:-$	$0:-$ $1:0$

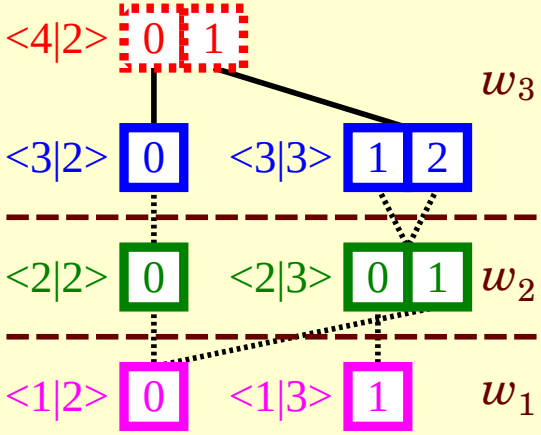
$\mathcal{S}_4 = \{\boxed{p^1}, \boxed{p^0}\} \equiv \{\boxed{0}, \boxed{1}\}$

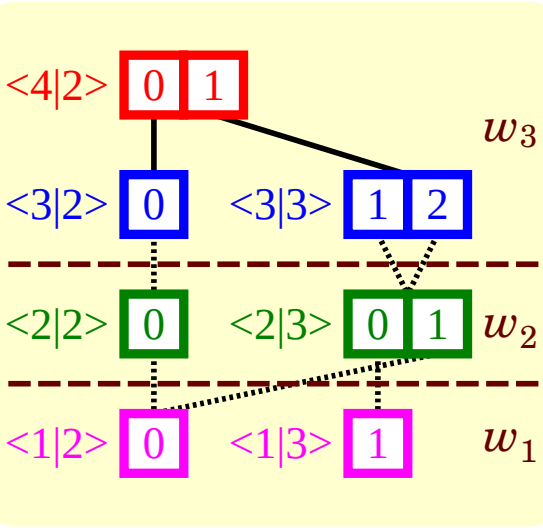
$\mathcal{S}_3 = \{\boxed{q^0r^0}, \boxed{q^1r^0}, \boxed{q^0r^1}\} \equiv \{\boxed{0}, \boxed{1}, \boxed{2}\}$

$\mathcal{S}_2 = \{\boxed{s^0}, \boxed{s^1}\} \equiv \{\boxed{0}, \boxed{1}\}$

$\mathcal{S}_1 = \{\boxed{t^0}, \boxed{t^1}\} \equiv \{\boxed{0}, \boxed{1}\}$

Saturate $\langle 4|2 \rangle$ on w_3 : local and remote firing of e





a	$\{b, c\}$	d	e
$0 : 1$ $1 : -$	**I**	**I**	$0 : -$ $1 : 0$
$0 : 1$ $1 : -$ $2 : -$	$0 : -$ $1 : 2$ $2 : 1$	**I**	$0 : -$ $1 : -$ $2 : 0$
$0 : 1$ $1 : -$	**I**	$0 : -$ $1 : 0$	**I**
I	**I**	$0 : 1$ $1 : -$	$0 : -$ $1 : 0$

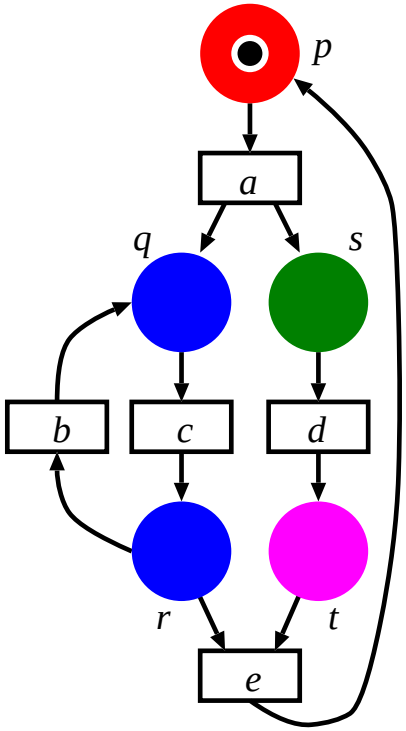
$$\mathcal{S}_4 = \{\boxed{p^1}, \boxed{p^0}\} \equiv \{\boxed{0}, \boxed{1}\}$$

$$\mathcal{S}_3 = \{\boxed{q^0 r^0}, \boxed{q^1 r^0}, \boxed{q^0 r^1}\} \equiv \{\boxed{0}, \boxed{1}, \boxed{2}\}$$

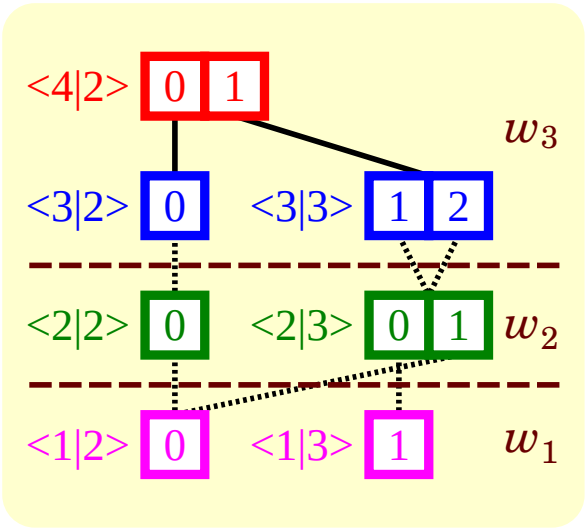
$$\mathcal{S}_2 = \{\boxed{s^0}, \boxed{s^1}\} \equiv \{\boxed{0}, \boxed{1}\}$$

$$\mathcal{S}_1 = \{\boxed{t^0}, \boxed{t^1}\} \equiv \{\boxed{0}, \boxed{1}\}$$

Final



a	$\{b, c\}$	d	e
$0 : 1$ $1 : -$	**I**	**I**	$0 : -$ $1 : 0$
$0 : 1$ $1 : -$ $2 : -$	$0 : -$ $1 : 2$ $2 : 1$	**I**	$0 : -$ $1 : -$ $2 : 0$
$0 : 1$ $1 : -$	**I**	$0 : -$ $1 : 0$	**I**
I	**I**	$0 : 1$ $1 : -$	$0 : -$ $1 : 0$



$$\mathcal{S} = \left\{ \begin{matrix} 0 \\ 0 \\ 0 \\ 0 \end{matrix}, \begin{matrix} 1 \\ 1 \\ 0 \\ 1 \end{matrix}, \begin{matrix} 1 \\ 1 \\ 1 \\ 0 \end{matrix}, \begin{matrix} 1 \\ 2 \\ 0 \\ 1 \end{matrix}, \begin{matrix} 1 \\ 2 \\ 1 \\ 0 \end{matrix} \right\}$$

Dynamic Memory Load Balancing

Dynamic memory load balancing

- **Problem** :

  With a static MDD level allocation, distributed state-space generation might fail because of excessive memory requirements at a single workstation

- **Solution** :

  Dynamic memory load balancing to utilize the resources of each workstation

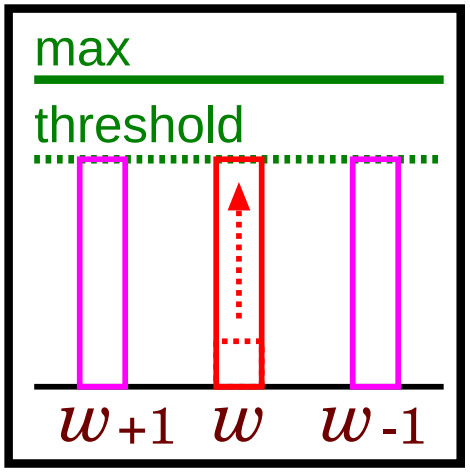
  - We cope with each workstation's *dynamic peak space requirement* by reallocating levels between neighboring workstations
  - An *optimal* static MDD level allocation could still be inferior to a *sub-optimal dynamic approach*
  - No need to exchange global memory consumption

Exchanging information about memory consumption

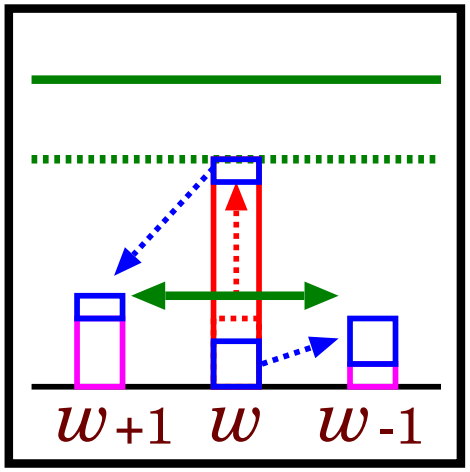
- w *periodically* monitors its **memory usage** and sends it to $w+1$ and $w-1$

- With a *single memory threshold*, each w
  - sets a memory **threshold** according to the **initial free space**
  - performs one of the following, if its **memory usage** is over the **threshold**

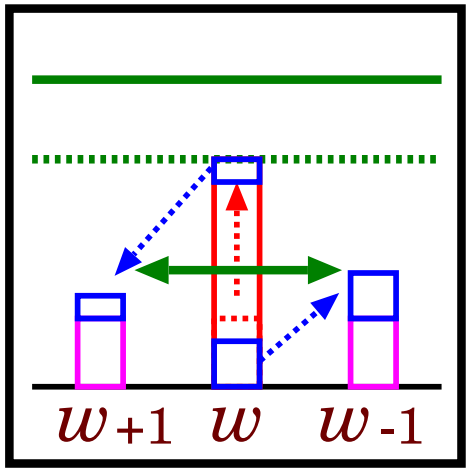
Abort



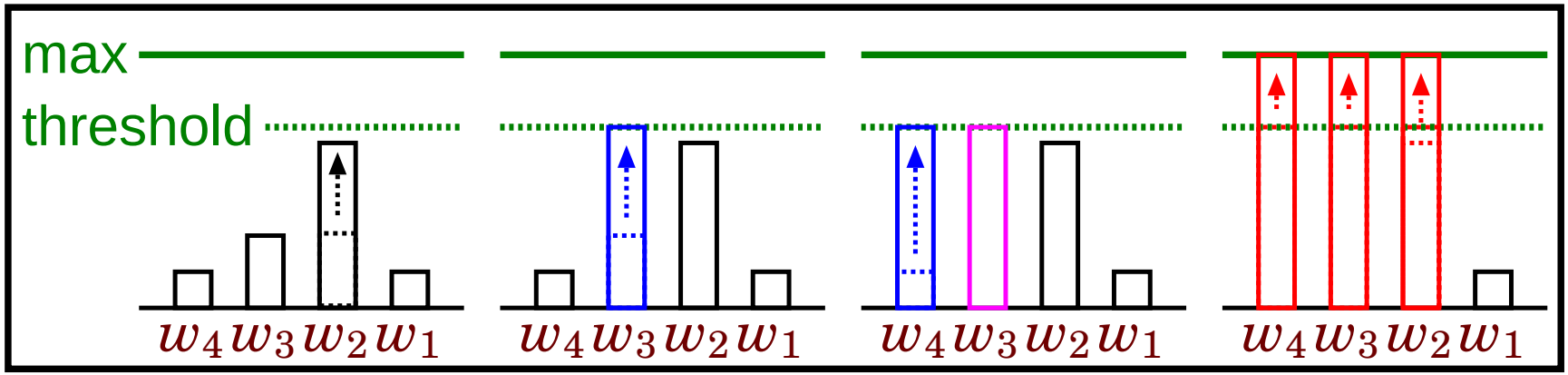
Send $mybot$ to $w-1$



Send $mytop$ to $w+1$

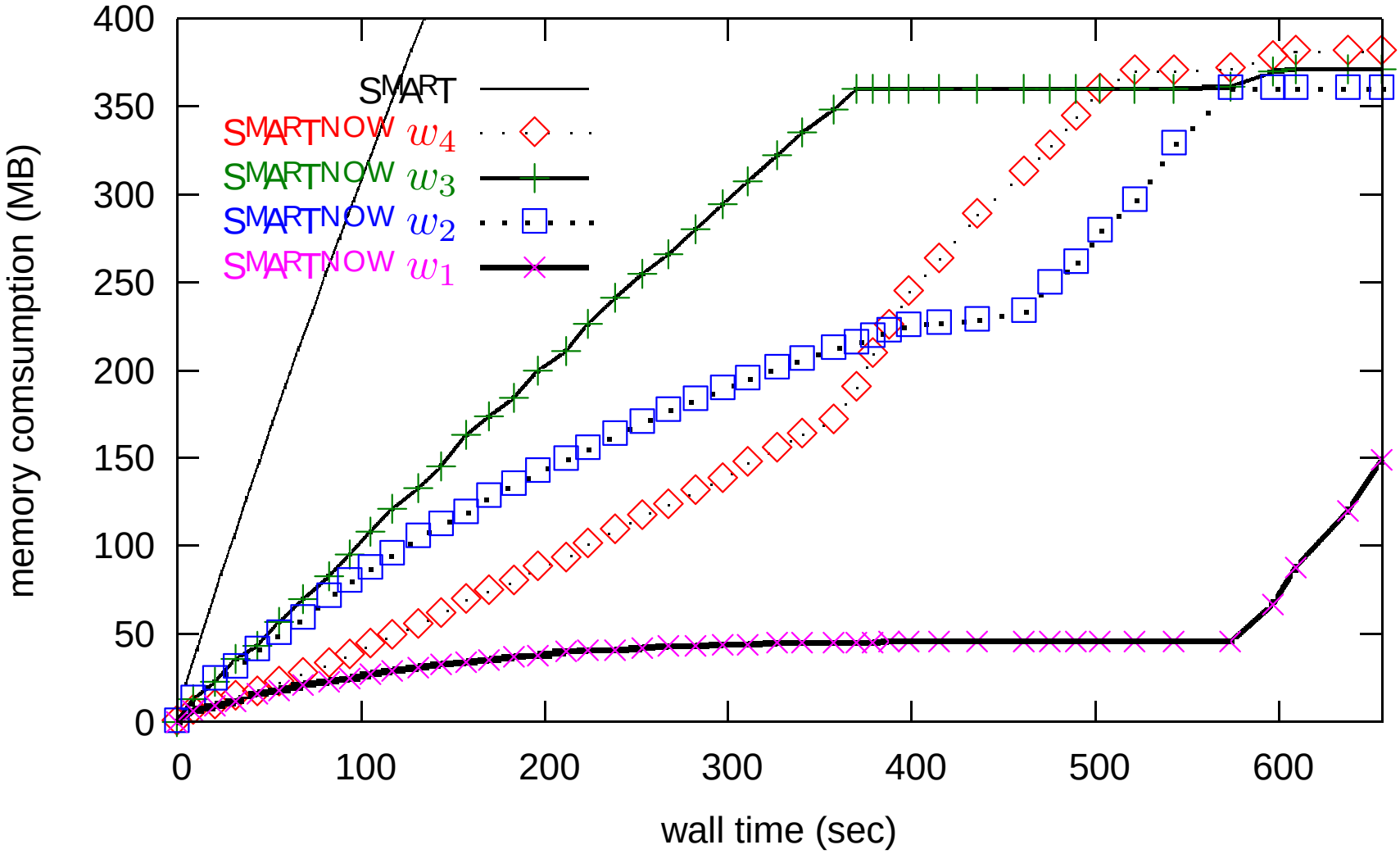


Pairwise memory load balancing dilemmas

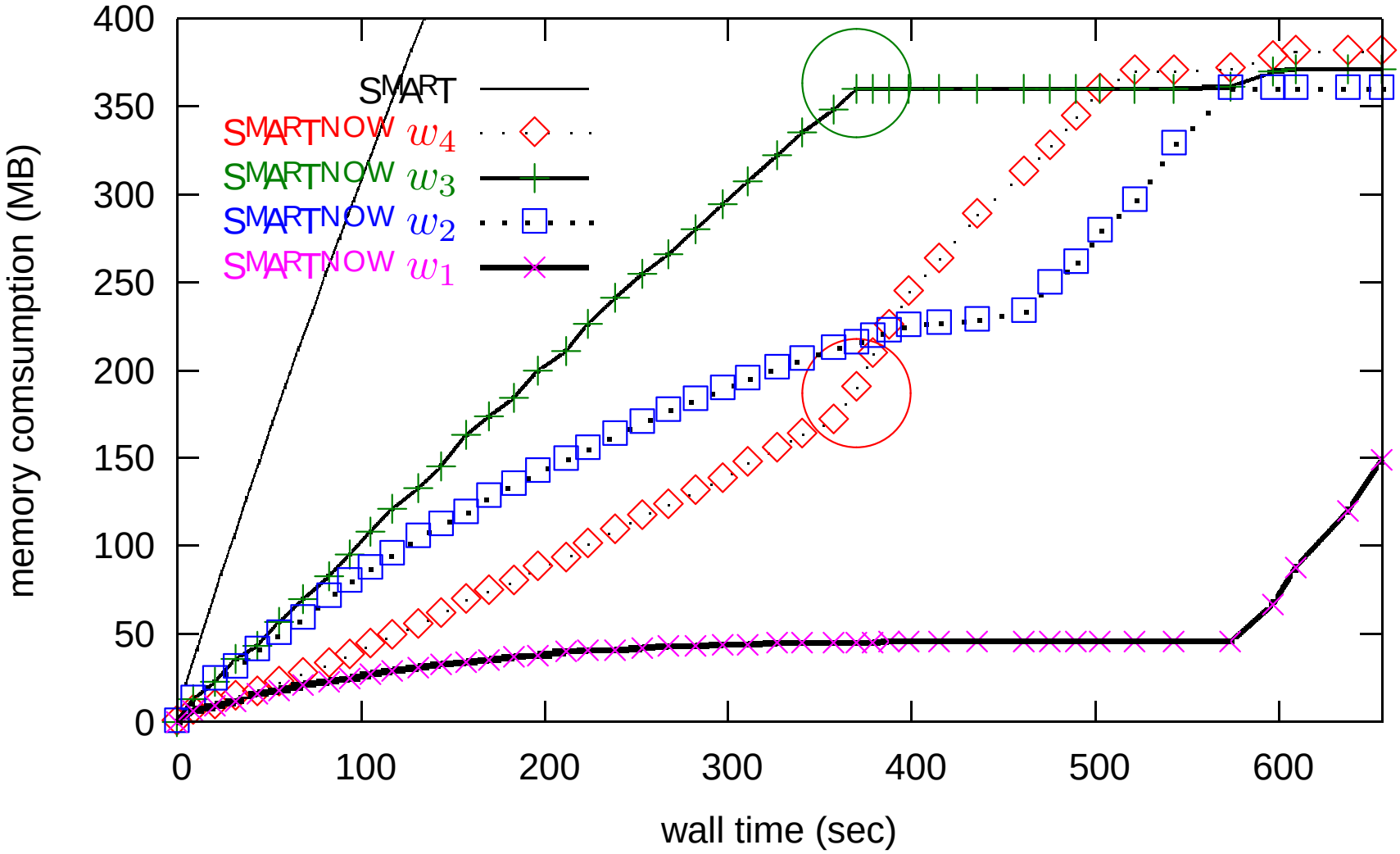


- Issues in memory load balancing
  - **Pairwise method** might not utilize the overall NOW memory well
  - **Global method** requires broadcast or global synchronization
- *Nested* memory load balancing :
  - Allows workstations trigger memory load balancing procedure while performing memory load balancing
  - Utilizes the overall NOW memory without knowing *global memory consumption*

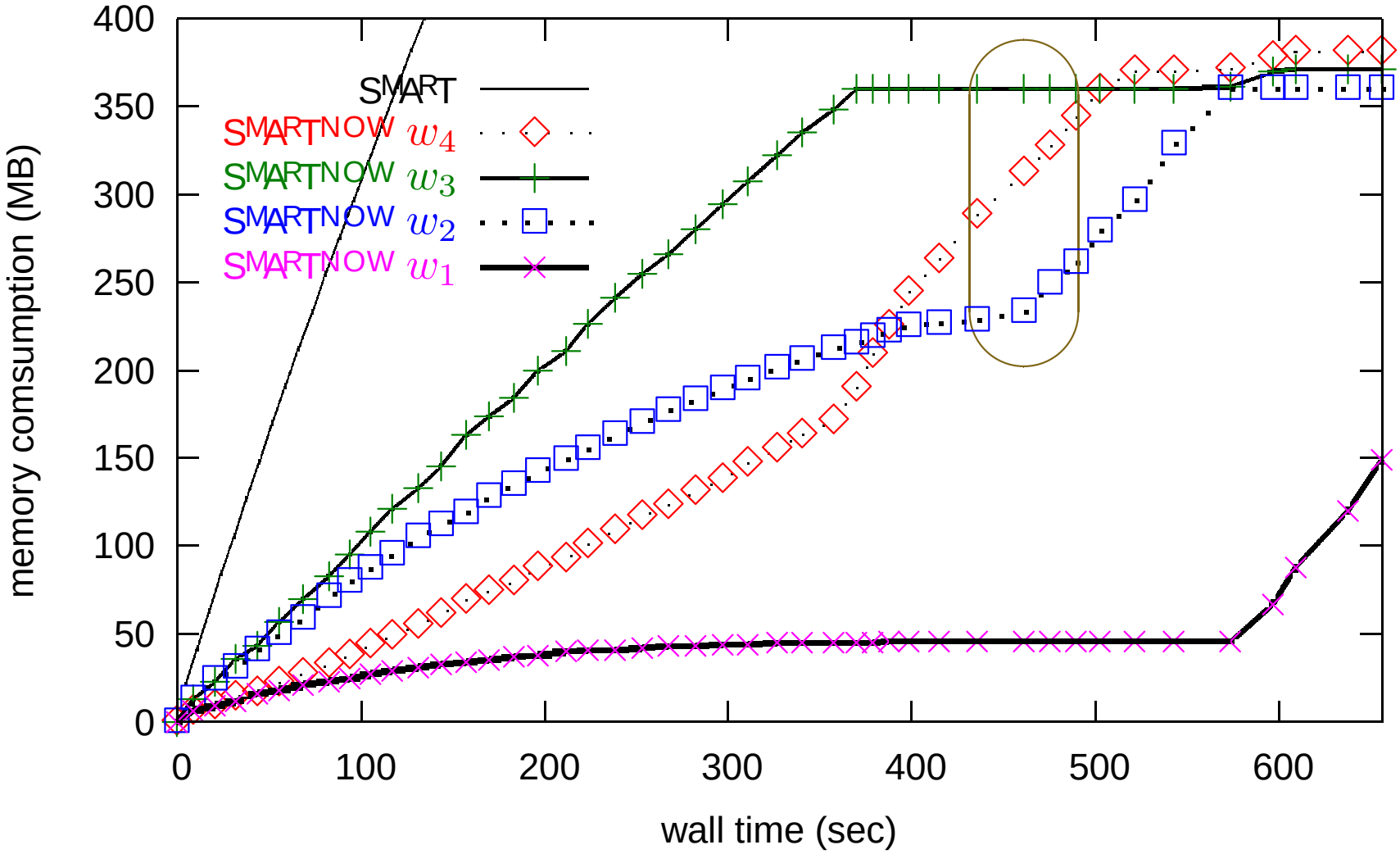
Nested memory load balancing case study



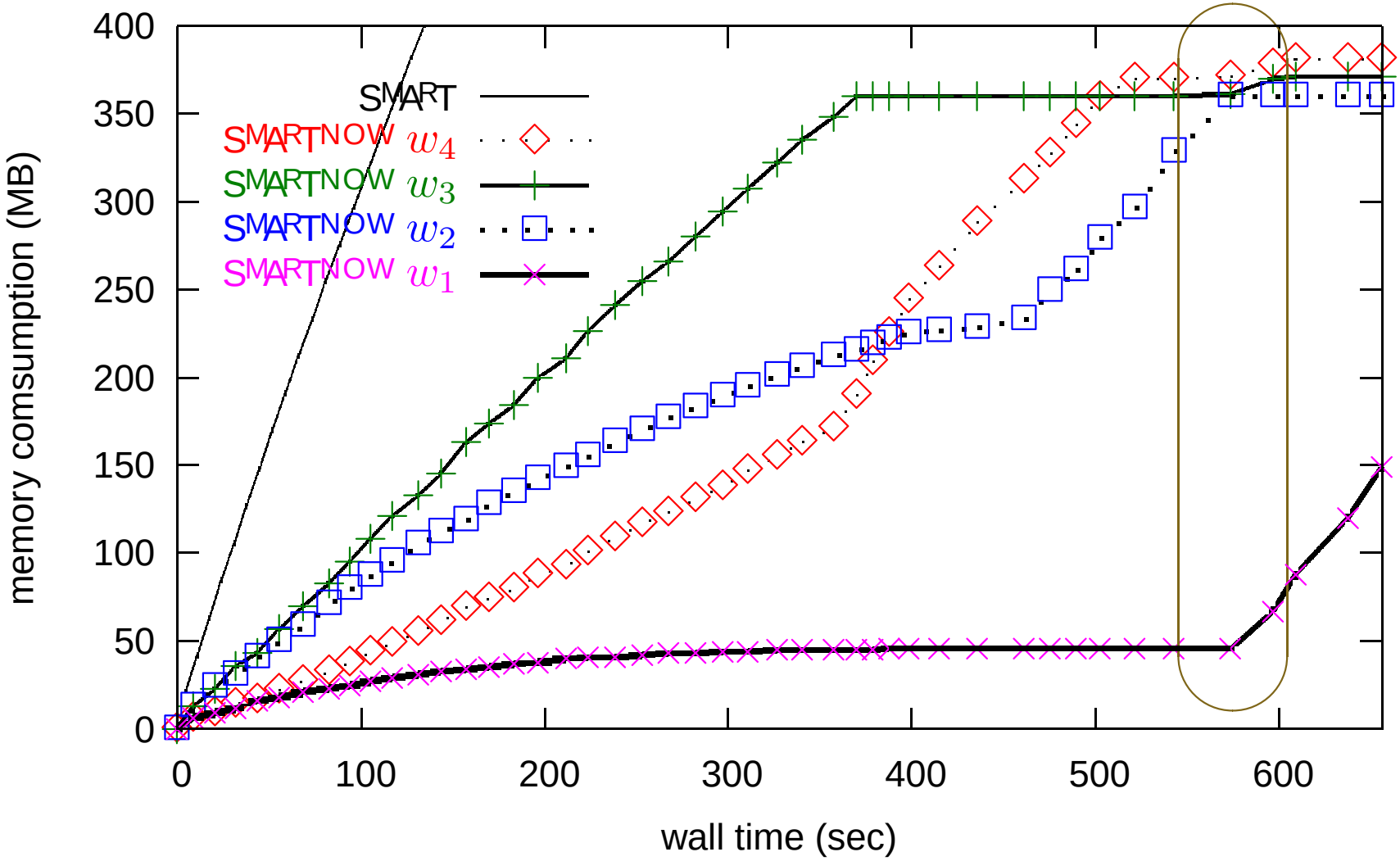
Nested memory load balancing case study



Nested memory load balancing case study



Nested memory load balancing case study



Experimental Results and Conclusions

SMART vs. SMARTNOW case study 1

- **Round robin mutex protocol with N processes**

$K = N + 1$, $|\mathcal{S}_k| = 10$ for all k except $|\mathcal{S}_1| = N + 1$, $|\mathcal{E}| = 5N$

Benchmark			Runtime (seconds)									
			SMART	SMARTNOW 2 @ 512MB			SMARTNOW 4 @ 512MB			SMARTNOW 8 @ 512MB		
N	$\|\mathcal{S}\|$	MB	512MB	Gigabit	100M	10M	Gigabit	100M	10M	Gigabit	100M	10M
300	$1.37 \cdot 10^{93}$	71	3	4	6	11	7	9	23	13	15	35
450	$2.94 \cdot 10^{138}$	158	8	9	11	19	12	15	38	20	31	63
600	$5.60 \cdot 10^{183}$	280	14	16	20	35	18	25	32	30	37	94
750	$9.99 \cdot 10^{228}$	434	21	24	29	58	29	39	113	41	55	130

- If RAM is *sufficient*, SMART runtime is always less than SMARTNOW runtime
- Message-passing overhead is *not trivial* but *not huge* either
- Difference between SMART and SMARTNOW $W = 8$ *diminishes* as N grows

SMART vs. SMARTNOW case study 1

- *Round robin mutex protocol* with N processes

$K = N + 1$, $|\mathcal{S}_k| = 10$ for all k except $|\mathcal{S}_1| = N + 1$, $|\mathcal{E}| = 5N$

Benchmark			Runtime (seconds)									
			SMART	SMARTNOW 2 @ 512MB			SMARTNOW 4 @ 512MB			SMARTNOW 8 @ 512MB		
N	$\|\mathcal{S}\|$	MB	512MB	Gigabit	100M	10M	Gigabit	100M	10M	Gigabit	100M	10M
300	$1.37 \cdot 10^{93}$	71	3	4	6	11	7	9	23	13	15	35
450	$2.94 \cdot 10^{138}$	158	8	9	11	19	12	15	38	20	31	63
600	$5.60 \cdot 10^{183}$	280	14	16	20	35	18	25	32	30	37	94
750	$9.99 \cdot 10^{228}$	434	21	24	29	58	29	39	113	41	55	130

- **If RAM is sufficient, SMART runtime is always less than SMARTNOW runtime**
- Message-passing overhead is *not trivial* but *not huge* either
- Difference between SMART and SMARTNOW $W = 8$ *diminishes* as N grows

SMART vs. SMARTNOW case study 1

- *Round robin mutex protocol* with N processes

$K = N + 1$, $|\mathcal{S}_k| = 10$ for all k except $|\mathcal{S}_1| = N + 1$, $|\mathcal{E}| = 5N$

Benchmark			Runtime (seconds)									
			SMART	SMARTNOW 2 @ 512MB			SMARTNOW 4 @ 512MB			SMARTNOW 8 @ 512MB		
N	$\|\mathcal{S}\|$	MB	512MB	Gigabit	100M	10M	Gigabit	100M	10M	Gigabit	100M	10M
300	$1.37 \cdot 10^{93}$	71	3	4	6	11	7	9	23	13	15	35
450	$2.94 \cdot 10^{138}$	158	8	9	11	19	12	15	38	20	31	63
600	$5.60 \cdot 10^{183}$	280	14	16	20	35	18	25	32	30	37	94
750	$9.99 \cdot 10^{228}$	434	21	24	29	58	29	39	113	41	55	130

- If RAM is *sufficient*, SMART runtime is always less than SMARTNOW runtime
- **Message-passing overhead is not trivial but not huge either**
- Difference between SMART and SMARTNOW $W = 8$ *diminishes* as N grows

SMART vs. SMARTNOW case study 1

- *Round robin mutex protocol* with N processes

$K = N + 1$, $|\mathcal{S}_k| = 10$ for all k except $|\mathcal{S}_1| = N + 1$, $|\mathcal{E}| = 5N$

Benchmark			Runtime (seconds)									
			SMART	SMARTNOW 2 @ 512MB			SMARTNOW 4 @ 512MB			SMARTNOW 8 @ 512MB		
N	$\|\mathcal{S}\|$	MB	512MB	Gigabit	100M	10M	Gigabit	100M	10M	Gigabit	100M	10M
300	$1.37 \cdot 10^{93}$	71	3	4	6	11	7	9	23	13	15	35
450	$2.94 \cdot 10^{138}$	158	8	9	11	19	12	15	38	20	31	63
600	$5.60 \cdot 10^{183}$	280	14	16	20	35	18	25	32	30	37	94
750	$9.99 \cdot 10^{228}$	434	21	24	29	58	29	39	113	41	55	130

- If RAM is *sufficient*, SMART runtime is always less than SMARTNOW runtime
- Message-passing overhead is *not trivial* but *not huge* either
- **Difference between SMART and SMARTNOW $W = 8$ diminishes as N grows**

SMART vs. SMARTNOW case study 1 cont.

- **Once swapping is triggered, SMART runtime increases dramatically**
- It can be *orders of magnitude larger* than that of SMARTNOW
- The *optimal number* W of workstations, W_{opt}, cannot be known a priori
- The penalty when $W < W_{opt}$ is *larger* than when $W > W_{opt}$

Benchmark			⋆ swapping occurred (time for DLMB) runtime in seconds									
			SMART	SMARTNOW 2 @ 512MB			SMARTNOW 4 @ 512MB			SMARTNOW 8 @ 512MB		
N	$\|\mathcal{S}\|$	MB	512MB	Gigabit	100M	10M	Gigabit	100M	10M	Gigabit	100M	10M
750	$9.99 \cdot 10^{228}$	434	21	24	29	58	29	39	113	41	55	130
900	$1.71 \cdot 10^{274}$	621	⋆ 928	(8) 41	101	558	40	51	152	54	71	302
1050	$2.85 \cdot 10^{319}$	870	⋆ 36751	⋆ (15) 72	501	2713	(5) 56	148	481	68	96	376
1200	$4.64 \cdot 10^{364}$	1201	—	⋆ (23) 368	1169	6610	(45) 91	1542	5410	87	143	472
1350	$4.76 \cdot 10^{409}$	1503	—	—	—	—	* (98) 196	3815	10754	(3) 112	1099	2245

SMART vs. SMARTNOW case study 1 cont.

- Once *memory swapping* is triggered, SMART runtime *increases dramatically*
- **It can be orders of magnitude larger than that of SMARTNOW**
- The *optimal number* W of workstations, W_{opt}, cannot be known a priori
- The penalty when $W < W_{opt}$ is *larger* than when $W > W_{opt}$

Benchmark			⋆ swapping occurred (time for DLMB) runtime in seconds									
			SMART	SMARTNOW 2 @ 512MB			SMARTNOW 4 @ 512MB			SMARTNOW 8 @ 512MB		
N	$\|\mathcal{S}\|$	MB	512MB	Gigabit	100M	10M	Gigabit	100M	10M	Gigabit	100M	10M
750	$9.99 \cdot 10^{228}$	434	21	24	29	58	29	39	113	41	55	130
900	$1.71 \cdot 10^{274}$	621	⋆ 928	(8) 41	101	558	40	51	152	54	71	302
1050	$2.85 \cdot 10^{319}$	870	⋆ 36751	⋆ (15) 72	501	2713	(5) 56	148	481	68	96	376
1200	$4.64 \cdot 10^{364}$	1201	—	⋆ (23) 368	1169	6610	(45) 91	1542	5410	87	143	472
1350	$4.76 \cdot 10^{409}$	1503	—	—	—	—	* (98) 196	3815	10754	(3) 112	1099	2245

SMART vs. SMARTNOW case study 1 cont.

- Once *memory swapping* is triggered, SMART runtime *increases dramatically*
- It can be *orders of magnitude larger* than that of SMARTNOW
- **The optimal number W of workstations, W_{opt}, cannot be known a priori**
- **The penalty when $W < W_{opt}$ is larger than when $W > W_{opt}$**

Benchmark			⋆ swapping occurred (time for DLMB) runtime in seconds									
			SMART	SMARTNOW 2 @ 512MB			SMARTNOW 4 @ 512MB			SMARTNOW 8 @ 512MB		
N	$\|\mathcal{S}\|$	MB	512MB	Gigabit	100M	10M	Gigabit	100M	10M	Gigabit	100M	10M
750	$9.99 \cdot 10^{228}$	434	21	24	29	58	29	39	113	41	55	130
900	$1.71 \cdot 10^{274}$	621	⋆ 928	(8) 41	101	558	40	51	152	54	71	302
1050	$2.85 \cdot 10^{319}$	870	⋆ 36751	⋆ (15) 72	501	2713	(5) 56	148	481	68	96	376
1200	$4.64 \cdot 10^{364}$	1201	—	⋆ (23) 368	1169	6610	(45) 91	1542	5410	87	143	472
1350	$4.76 \cdot 10^{409}$	1503	—	—	—	—	⋆ (98) 196	3815	10754	(3) 112	1099	2245

SMART vs. SMARTNOW case study 2

- **Slotted ring network model with N nodes**

  $K = N$, $|\mathcal{S}_k| = 15$ for all k, $|\mathcal{E}| = 3N$

Benchmark			⋆ swapping occurred (time for DLMB) runtime in seconds									
			SMART	SMARTNOW 2 @ 512MB			SMARTNOW 4 @ 512MB			SMARTNOW 8 @ 512MB		
N	$\|\mathcal{S}\|$	MB	512MB	Gigabit	100M	10M	Gigabit	100M	10M	Gigabit	100M	10M
100	$2.60 \cdot 10^{105}$	36	12	16	16	22	24	25	45	42	43	75
150	$4.53 \cdot 10^{168}$	125	44	50	52	61	64	68	115	91	97	152
200	$8.38 \cdot 10^{211}$	286	108	119	121	137	139	143	193	182	199	287
250	$1.59 \cdot 10^{265}$	545	⋆ 392	(9) 248	274	508	267	298	340	337	371	491
275	$7.04 \cdot 10^{291}$	737	⋆ 57414	(26) 355	486	1098	358	368	441	443	449	621
300	$3.11 \cdot 10^{318}$	962	—	⋆ (29) 552	2663	12457	(23) 490	1981	7333	564	589	763
325	$1.38 \cdot 10^{345}$	1213	—	⋆ (668) > 1d	> 1d	> 1d	(80) 656	9082	26199	716	743	941
350	$6.15 \cdot 10^{371}$	1588	—	—	—	—	⋆ (238) 1148	46910	> 1d	878	919	2470

- *Gigabit* network does help decrease the message passing overhead

SMART vs. SMARTNOW case study 2

- *Slotted ring network model* with N nodes

  $K = N$, $|\mathcal{S}_k| = 15$ for all k, $|\mathcal{E}| = 3N$

Benchmark			⋆ swapping occurred (time for DLMB) runtime in seconds									
			SMART	SMARTNOW 2 @ 512MB			SMARTNOW 4 @ 512MB			SMARTNOW 8 @ 512MB		
N	$\|\mathcal{S}\|$	MB	512MB	Gigabit	100M	10M	Gigabit	100M	10M	Gigabit	100M	10M
100	$2.60 \cdot 10^{105}$	36	12	16	16	22	24	25	45	42	43	75
150	$4.53 \cdot 10^{168}$	125	44	50	52	61	64	68	115	91	97	152
200	$8.38 \cdot 10^{211}$	286	108	119	121	137	139	143	193	182	199	287
250	$1.59 \cdot 10^{265}$	545	⋆ 392	(9) 248	274	508	267	298	340	337	371	491
275	$7.04 \cdot 10^{291}$	737	⋆ 57414	(26) 355	486	1098	358	368	441	443	449	621
300	$3.11 \cdot 10^{318}$	962	—	⋆ (29) 552	2663	12457	(23) 490	1981	7333	564	589	763
325	$1.38 \cdot 10^{345}$	1213	—	⋆ (668) > 1d	> 1d	> 1d	(80) 656	9082	26199	716	743	941
350	$6.15 \cdot 10^{371}$	1588	—	—	—	—	⋆ (238) 1148	46910	> 1d	878	919	2470

- **Gigabit network does help decrease the message passing overhead**

Conclusions and future work

- Saturation NOW is still a **sequential** algorithm
- SMARTNOW is able to solve models that are still not tractable by SMART
- *Level-based slicing* scheme performs better than the *job-based slicing* approach in terms of distributed memory utilization
- *Gigabit network* makes *level-based* slicing worth considering again
- With the nested DMLB scheme, SMARTNOW can *utilize the overall NOW memory with pairwise communication only*
- Possible approaches to *parallelize the saturation algorithm*

Thank You!